**Ch1~5 final**

1. 程式輸出資料至disk時，buffer如何運作

1 buffer = 1 physical record = N logical records

當CPU讀取某一個邏輯紀錄時，區塊設備會將包含此邏輯紀錄的實體紀錄讀入緩衝區內，CPU至緩衝區取到此邏輯紀錄；當CPU讀取第二個邏輯紀錄時，只要到緩衝區內取邏輯紀錄。

1. 排程的資訊如下，在分時系統(Time Sharing)環境下，當時間片段(Time Slice)為2時，請繪出排程的甘特圖(Gantt Chart)。

Process Burst Time Arrival Time

P1 11 0

P2 2 2

P3 3 3

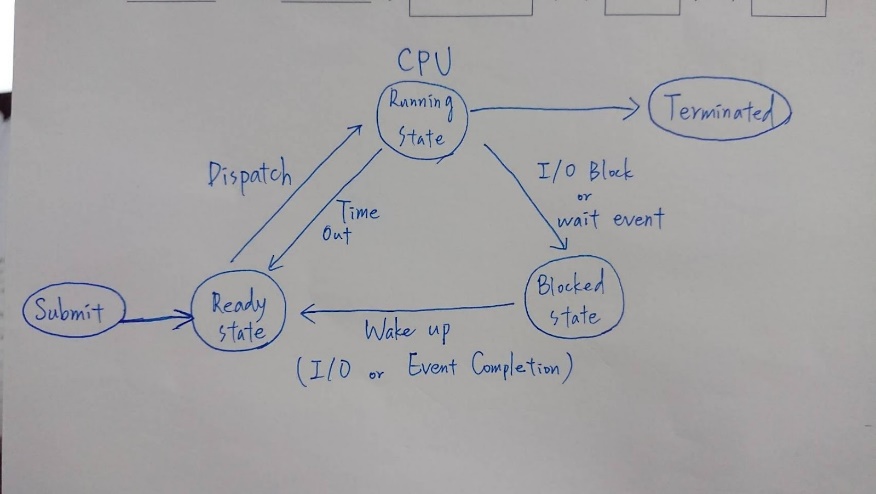
P4 2 4

P5 6 5

A close-up of a paper with writing

Description automatically generated

1. Context switch有哪三種情況，說明、畫圖



Running state -> Terminated：目前的處理元作完則換下一個處理元佔有CPU。

Running state -> Blocked state：目前的處理元再作I/O或是等待某事件發生，也換下一個處理元佔有CPU。

Running state -> Ready state：Time out時，下一個處理元佔有CPU。

1. Process和thread關係為何

每個Process裡有一個或多個Thread，裡面的這些Thread共用Code、Data、Resource，並各自擁有自己的PC、Register、Stack。

1. fork()與execve()有何不同

fork()產生子處理元後，父處理元與子處理元都會存在；execve()則是父處理元會結束。

1. Interrupt時OS如何處理

-> 控制權交給OS

-> OS save CPU status

-> OS判斷是哪一種中斷，並提供相對應的服務

-> 當中斷執行完畢，並restore CPU status

-> PC指向下一道指令

1. 現代作業系統大都採用何種中央處理器排程(CPU Scheduling)？請說明該排程的運作方式

Multilevel Feedback Queue。此種排程法有多個queue，process依生成時給予優先等級。順序上，以分時方式由較高優先等級的queue先執行，該queue執行完，則至次高優先等級的queue內選取處理程序進行分時處理。在某個queue內處理時，若較高優先等級的queue內有處理程序，則回至較高優先等級queue，以分時方式處理。允許處理程序依時間升級機制，提升至較高優先等級或降低至較低等級queue中。

1. 二部電腦如何透過Socket來通訊

雙方互相確認address和port number

**Ch6 final**

1. 使用號誌(Semaphore)及WAIT(S)、SIGNAL(S)是否滿足進入CS的三個條件

條件：互斥、行進、有限等待

互斥：呼叫Wait(S)時，看S之值決定能否進入臨界區間，以及將S值減1這二個動作是不會被中斷，所以保證只有一個處理程序能進入臨界區間；呼叫Signal(S)時，看S之值確認是否有處理程序等待進入臨界區間，如果有，就讓Queue內最前面的處理程序進入臨界區間，且將S值加1，這些動作是不會被中斷，保證最多只叫醒一個處理程序進入臨界區間。且因為有Queue的機制決定處理元進入臨界區間的次序，符合行進與有限等待。

1. 為何使用號誌、臨界區域(Critical Region) 及監督器(Monitor)還是有可能發生時間相依錯誤

因為可以用臨界區域或監督器寫程式語言的function來實現wait()、signal()，把實作的wait()、signal()當作號誌的wait()、signal()使用，能保護互斥，若誤用仍會產生時間相依錯誤。

1. 請比較號誌、臨界區域(Critical Region)、及監督器(Monitor)

process使用號誌時，wait( )及signal( )必須由程式撰寫者親自寫在程式中，容易造成時間相依錯誤問題。使用號誌是各做各的，但彼此利用號誌來達到互斥同步的目的。

monitor則總管其內部的程序及資料，任何使用其內部程序或資料的process都必須經過monitor總管及安排，達到互斥需求。

critical region使用了programming的語法,將要被保護的資料用{ and }宣告,之後使用compiler檢查語法,如果{ and }使用不當的,在編譯過程的時候會產生語法錯誤而不能使用。若語法無誤，則將{ and }翻成wait( ) and signal( )。

1. 寫出號誌(Semaphore)的WAIT(S)及SIGNAL(S)這二個最細小運算A screenshot of a computer

   Description automatically generated
2. 使用CS時撰寫程式比使用號誌更佳，請說明為什麼

當Process使用號誌，是自行寫wait and signal，但彼此利用號誌來達到進入臨界區間互斥和同步的目的，有可能誤用wait and signal而造成時間相依錯誤的問題。Critical region使用了programming的語法，將要保護的資料用{、}包起來，在經過compiler看語法是否有錯誤。若正確再將{、}變成wait及signal，語法能夠解決部分時間相依錯誤的問題。

1. 循序執行(Serial Execution)與可循序化執行(Serializable Execution))有何不同？有何相同？舉一個可循序化執行的例子。

循序執行：當一個process處理完後依序換下一個process執行

可循序化執行：若該處理器使用另一種排程，而結果與循序執行相同，則稱processes是可循序化執行的

執行process1 read(A) write(A)，再執行process2 read(A) write(A)，接著執行process1 read(B) write(B)，然後執行process2 read(B) write(B)

A blue and white rectangular boxes with text

Description automatically generated with medium confidence

1. 何謂號誌

是一種保護變數，藉由P和V來做運算。P和V運算是細小、不可被中斷的處理。

1. 使用Critical Region與使用號誌有什麼不同

號誌必須自己加上wait()跟signal()來保護互斥資源

臨界區域可使用PL的{}語法來保護互斥資源

1. 何謂臨界區間(Critical Section)？處理元進入臨界區間要考量什麼問題

process去存取共用資源的那段code。

要考慮互斥、行進、有限等待。

1. 有一個test-and-set( )函數(Function)，它是利用硬體來實現，它的內容如下所示，進入臨界區間的程式亦同時撰寫在下面。為何此進入臨界區間的程式僅滿足互斥，而不滿足其他二條件？

A white background with black text

Description automatically generated

看lock是否為true及將lock改為true這二件事是不會被中斷，所以僅會有一個處理程序進入臨界區間。

若同時有兩個處理元Pi Pj要求進入臨界區間，程式沒有辦法去處理當Pi離開，要換Pj進去，沒有使用queue的排隊機制，不符合行進，也不符合有限等待

1. 有一個演算法如下，請問它符合進入臨界區間(Critical Section)三條件的那些條件？請證明。A white background with black text

   Description automatically generated

互斥 : 若兩個process僅有一個flag為true，則這個flag為true的process可以進入critical section，另一個process不須進入critical section，因此他們互斥。

若兩個process都要求進入critical section，且各別的flag也是true，因為turn值指可能是i或j，若turn=i則Pi進入critical section，Pj在迴圈等待；反之。

行進 : 當Pi或Pj均要求進入critical section，若Pi進入critical section，Pj在迴圈等待。只要Pi離開critical section時，它會將flag[i]設為false，turn=j，因此Pj便可以進入critical section，這符合行進的條件。因此只要對方從critical section出來，另一方就可以進入critical section。

有限等待 : Pj只要等待Pi進入critical section一次，它便可以進入critical section。

1. 可循序化執行(Serializable Execution)次序，到底有那些是可以的
2. 證明Bakery演算法滿足進入臨界區間(Critical Section)的三個條件

互斥->只有號碼牌最小的process可以進臨界區間，若號碼牌一樣小，以pid小決定。

行進->以抽號碼來排隊，每次有process離開臨界區間，就換下個最小編號的process進入。

有限等待->有號碼牌為依據，假設有n個process，號碼牌最大的process頂多等n-1次就可執行到。

**Ch7 final**

1. 請問TA必須REDO還是UNDO？為什麼？

A screenshot of a computer

Description automatically generated

TA要REDO因為在記錄檔裡頭看到start TA且看到commit TA

，表示TA有完成，但可能有些資料還在緩衝區，所以只要根據交易紀錄檔內容，將交易紀錄trace back，並與磁碟內的資料做比對，若磁碟內資料不對，則修正。

1. 那些交易必須REDO？那些交易必須UNDO？為什麼？

A white background with black text

Description automatically generated

看到start跟commit -> REDO ； 看到start沒有commit -> UNDO。

REDO：TB

UNDO：TA TC

REDO TB 因為看到start且看到Commit，但可能有些資料在緩衝區尚未寫入磁碟，所以只要根據交易紀錄檔內容，將交易紀錄trace back，並與磁碟內的資料做比對，若磁碟內資料不對，則修正。UNDO TA TC因為看不到commit，表示沒有執行結束就當掉，將交易紀錄trace back直到此交易的開始，並將磁碟內已經修改的資料還原到此交易沒有執行前的狀況。

1. Banker‘s演算法如下所示，若P4要求3A3B1C時，請計算系統是否可以接受此要求？

(就算唄反正答案是不行，懶得貼題目了)

P4要求3A3B1C未超過Need。

給P4，P4的Need仍需要1A，所以P4未執行完成，Available剩0A0B1C，無法供其他處理元完成工作，因此不能允許P4的要求。

1. 四個死結必要條件

互斥：某資源被一個處理元使用時，其他處理元需等待此資源被釋放才能使用。

擁有和等待：處理元已經擁有至少一個資源，但仍需等待其他處理元擁有的資源。

不可奪取：若有資源被一個處理元擁有，其他處理元不能搶奪此資源，須等此資源被正常釋放。

循環等待：一組處理元P0, P1, ….., Pn，P0在等P1的資源，P1在等P2的資源，….，Pn在等P0的資源，稱為循環等待。

1. 寫哲學家用餐的code並說明甚麼情況會發生死結

wait(use[i]);

wait(use[i mod x]);

eat;

signal(use[i]);

signal (use[i mod x]);

think;

上述為x個哲學家中的第i位哲學家要用餐時的情境。當哲學家取得左邊筷子的時候，發生環境切換後，很有可能每位哲學家都取得左邊筷子了，之後切回來原本的哲學家可能就無法取得右邊筷子了，會造成死結。

1. 為何哲學家用餐問題符合死結(Deadlock)的四個必要條件(Necessary Conditions)

互斥：筷子這個資源是互斥的，當一個哲學家擁有筷子，他不允許別的哲學家擁有此筷子。

擁有和等待：當每個哲學家均擁有右邊的筷子，卻無法取得左邊的筷子，因為左邊的筷子已經被相鄰的哲學家拿走了。

不可奪取：每個哲學家均擁有右邊的筷子，卻堅持已經擁有的筷子不能被別的哲學家奪取。

循環等待：哲學家1等待哲學家2的筷子、哲學家2等待哲學家3的筷子、哲學家3等待哲學家1的筷子，產生循環等待

1. 若要處理復原(Recovery)問題，請問交易紀錄檔(Transaction Log File)內要記錄那些內容

紀錄修改的命令

修改前跟修改後的資料值

1. 安全狀態、不安全狀態與死結的關係ch7 p.27

在安全狀態絕對不會產生死結

不安全狀態不一定會產生死結

死結一定是從不安全狀態中產生

1. 預防死結的方法是讓死結的某一個必要條件不成立，請問實務上可行嗎？為什麼？

不可行

互斥：不可共用資源必須維持互斥性。

不可奪取：某些是專屬設備，無法打破不可奪取。

循環等待：理論上可以給予每個資源一個唯一的編號，處理元可以再任意時間要求資源，但每次所要求資源的編號必須漸增。實務上不可行因為資源有幾個無法預先知道。

擁有和等待可行但不能用：會花很多時間，可能會餓死。

1. 請問電腦系統必須提供那些機制，以備電腦當掉後，能進行復原(Recovery)的工作？

使用記錄檔來記錄每個處理元的執行過程，包含

記錄修改的命令（Modify Command）

修改前的資料值（Before Image）

修改後的資料值（After Image）

1. 若檢查點演算法(Checkpoint Algorithm)的Step 1與Step 2次序執行反了，會造成什麼問題

如果在執行step1時當掉 重新啟動 資料已被寫入但是 修改命令尚未寫入不知道是做了甚麼command來修改的

1. 若作業系統偵測到處理元發生死結，請問如何處理？是否會有後遺症發生？

選擇一個犧牲者，將此犧牲者回復到沒有執行前的狀態。萬一消除一個process仍無法解開死結，則持續消除另一個process直到死結解開。

解開死結並不能保證某些處理元不會餓死。

1. 說明時間戳記演算法

任何process在執行時，會有一個時間戳記，每個資源都有讀取時間戳記R-timestamp(O)及寫入時間戳記W-timestamp(O)。當某個process欲讀取或寫入資源時，依下列演算法處理。

**Ch8 final**

1. 請說明分頁法(Paging)的運作方式，並說明此方法是否發生何種支離破碎(Fragmentation)問題？

將secondary memory的program依固定大小切成page，並將main memory依一樣的大小切出page frame，若當下要執行的code不在page frame中則發生page fault，並找到一個沒再使用的page frame放進去，若沒有則依當下使用的演算法進行page replacement

分頁法還是會發生內部支離破碎。

1. 假設分頁法(Paging)一個頁框(Page Frame)的大小為256 Bytes，若一個邏輯地址(Logical Address)為76543210，請問此邏輯地址的頁碼(Page Number)是多少？若該邏輯地址是使用8個Nibbles代表，則虛擬記憶體有多大

(1) 256 B = 2^8

8 bits = 2 nibbles

Page Number = 765432

(2) 8 nibbles = 32 bits

2^32 = 2^2 \* 2^10 \* 2^10 % 2^10 = 4 GB = 虛擬記憶體大小

1. 假設關聯記憶體(Associative Memory)之存取時間為10ns，主記憶體(Main Memory)之存取時間為200ns。若採用混合式關聯記憶體及主記憶體的頁表格架構，且在關聯記憶體內的命中比率(Hit Ratio)為90%，請問需要花多少時間找到該頁的實體地址

計算！！

1. 請說明使用關聯記憶體(Associative Memory)及主記憶體混合式的頁表格(Page Table)架構，如何進行運作？

先利用Search key去TLB尋找對應的頁框碼，若找不到則至表格內尋找，之後都整合成Physical Address。之後依據快取記憶體的演算法，決定是否要將頁框碼及頁碼放到TLB裡面，若TLB滿了，則利用Replacement的方法還置換。

1. 比較外部支離破碎(External Fragmentation)及內部支離破碎(Internal Fragmentation)

外部支離破碎將多個程式放入主記憶體中，程式之間有未使用的地方，這些未使用的地方總和大於將載入的程式大小，但是因為他分散，故無法使用

內部支離破碎則是將main memory分成固定分割的段落，程式載入固定段落後，每個段落勝的空間大於新載入的程式，稱為內部支離破落

兩者差異在於是否將main memory分成固定的段落。

1. 請說明分段法(Segmentation)的運作方式，並說明為何此方法不會發生外部支離破碎及內部支離破碎。

將輔助記憶體內的程式依邏輯功能切成segment，因此每個區段的大小是變動的，segment table要紀錄程式在主記憶體內的起點及大小

將logical address 轉 physical address :

(1)至segment table找到起點跟大小

(2)檢查位移是否超過大小，若沒有，則physical address為起點+位移

由於切segment時，是依照程式大小切的，所以不會有內部碎裂。

而每隔一段時間系統會做garbage collection，所以可以解決外部碎裂。

1. 可重定位分割法(Relocatable Partition)是虛擬記憶體管理(Virtual Memory Management)還是實體記憶體管理(Physical Memory Management)？它會有支離破碎(Fragment)問題嗎？為什麼？

實體記憶體管理

大塊支離破碎，仍可能有程式占有大塊記憶體卻很少被用到

1. 有一個分頁法(Paging)，其頁的大小為64KB，現有一個邏輯地址(Logical Address)為FFFFFFF321，請問此邏輯地址是在該程式的那個頁碼(Page Number)？為什麼？請問一個程式最大可以有多大？為什麼？

64KB = 2^6 \* 2^10 = 2^16

16 / 4 = 4 (Nibbles)

FFFFFF | F321

Page Number Offset

所以 Page Number在FFFFFF。因為可以從64KB算出4 Nibbles

Page Number FFFFFF 共有6個Nibbles，加上Offset F321 的4個Nibbles 共有10個Nibbles => 10 \* 4 = 40 => 240 = 1TB

1. 假設分頁法(Paging)一個頁的大小(Page Size)為64KB，若一個邏輯地址(Logical Address)為777777777，且頁表格(Page Table)如下所示，請問此邏輯地址對應的實體地址(Physical Address)為多少？為什麼？

A rectangular object with numbers

Description automatically generated

64KB = 2^6\*2^10 = 2^16 = 4 nibble

offset = 7777

所以實體位置為7777->567 + offset = 567 7777

1. Ch8 p.45吧

**Ch9 final**

1. 何謂工作集ch9 p.43

工作集為處理元在某段時間使用頁的集合。

用working set抓每一個process的locality，就可以給出足夠的page frame來避免產生page thrashing。

1. 說明Belady’s Anomaly，甚麼置換法會發生

增加框頁數，反而早成更多的頁錯誤及頁置換；先進先出頁置換法。

1. 頁緩衝區法(Page Buffering Algorithm)的運作方式

OS有一個free frame pool，發生page replace時，

從free frame pool中選出一個free frame，把miss page載入到此frame即可執行

挑出victim page

當I/O設備有空時，將victim page寫回disk，再將此frame放置free frame pool中

1. 頁緩衝區法比傳統的頁置換有較佳的效能，為什麼ch9p.32

傳統

1.找到犧牲者，搬離頁框並寫入輔助記憶體

2.將發生錯誤的那一頁由輔助記憶體載入頁框

3.執行頁框內的程式或資料

頁緩衝區

1.先將發生錯誤的那頁放入輔助記憶體內

2.將緩衝區的那頁併入主記憶體

3.執行頁框的程式片段或程式碼

4.將犧牲者寫入輔助記憶體

1. 局限性(Locality)、工作集(Work Set)及頁的來回動盪(Page Thrashing)之間的關係ch9 p.31

可以從工作集看出目前的局限性，因而知道需要多少page frame。依局限性給足夠的page frame，就不會造成page thrashing。

1. 侷限性是啥ch9 p.32

時間侷限性就是最近發生的事情在不久的將來再發生的可能性非常大，就main memory來說，某些記憶體位置最近被參考到，則不久還會再被參考到。

空間侷限性類似於時間侷限性，某個地方發生問題，則相鄰地方發生問題的機會非常的大。

1. 有一個程式如下所示,若陣列(Array)為列優先(Row Major),且 sizeof(integer)=2 Bytes。若頁的大小為 256 Bytes,且系統提供 4 個頁框供陣列使用,請問該程式執行完,在此 4 個頁框內共發生幾次頁錯誤及頁置換?

A number on a white background

Description automatically generated

Page Fault = (256/2) \* 64 = 642次

Page Replacement = (256/2) \* 64 - 4 = 642 - 4次

1. 就分頁法的角度，請舉一個例子說明程式結構影響頁錯誤(Page Fault)及頁置換(Page Replace)。

A number on a white background

Description automatically generated

若程式這樣寫，每當執行到A[i][j] = 0時，則會發生一次Page Fault，當Page Frame都被填滿時，除了Page Fault以外，還會發生Page Replacement。

1. 舉一個電腦內發生空間局限性(Spatial Locality)的例子，並說明為何此例子符合空間侷限性

程式內對array的存取，存取某一個元素，接著可能存取附近的元素

1. 若系統採用最近罕用頁置換法(LRU)，程式執行時使用的頁碼依序為1,2,3,4,1,2,5,1,2,3,4,5，且頁框(Page Frame)個數為4，請問共發生幾次頁錯誤(Page Faults)及頁置換(Page Replaces)，若視窗(Window)為時間0至時間5，則其工作集(Working Set)是什麼？

Page Faults 8 次，Page Replaces 4次

此Working Set有1,2,3,4,1,2 1,2,3,4為此Working Set找的locality

1. 何謂Page fault, 何謂Page replacement
2. Ch9 p.48()頁大小的理由

**Ch10 final**

1. 就檔案系統而言，開啟檔案的目的是什麼

將檔案的FCB從輔助記憶體複製到主記憶體的已開檔表格。

1. 畫一般圖形目錄結構
2. 非循環圖形目錄結構

多個目錄可以包含相同的子目錄或檔案，但這些子目錄或檔案只有一份。

A diagram of a computer program

Description automatically generated

1. Windows採用何種目錄結構

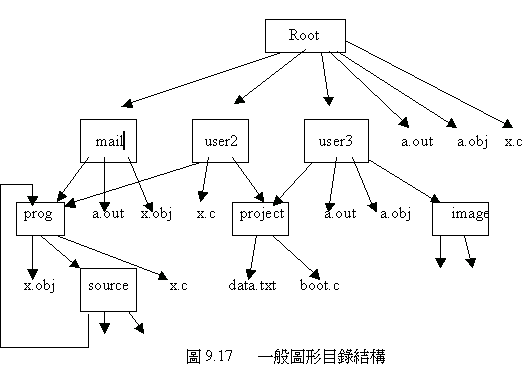
一般圖形結構。任何一個目錄，可以把已存在的目錄當作子目錄，而形成一個循環圈（Cycle）的圖形

A diagram of a root system

Description automatically generated

1. 說明unix的目錄結構(Directory Structure)，並說明這種目錄結構與MS/DOS的目錄結構有何不同

UNIX為一般圖形結構，其特點為任一個目錄可以把已存在的目錄當作子目錄，而形成cycle MS/DOS為樹狀圖形結構



1. 何謂翻轉檔案(Inverted File)

每一個索引鍵建立一個翻轉檔案，翻轉檔案內有索引鍵欄位及邏輯記錄編號欄位，並將索引鍵值排序。

用索引鍵值至翻轉檔案內進行二元搜尋(Binary Search)，找到符合的資料是那一個邏輯記錄編號，透過編號找到該筆資料。

1. 如何使用翻轉檔案(Inverted File)建立索引檔(Index File)

每個Search Key都會建立一個Inverted File，其中Inverted File包含了Search Key、Record Number。每個Inverted File會依照search key去sort、透過search key對 index file進行binary search、也透過search key去找到包含logical record的資料。

1. 從檔案系統(File System)的角度來看，為何使用檔案前必須打開檔案(Open file)？從讀寫(Read/Write)資料的角度來看，為何使用檔案前必須打開檔案？

將file control block複製一份到memory中，這樣要使用此file的資料時，才會比較快。

從讀寫(Read/Write)資料的角度來看，打開檔案是要進入臨界區間，必須維持共用資源互斥性。

**Ch11 final**

1. 請以此圖說明為何檔案配置表格(File Allocation Table, FAT)可以進行檔案空間配置(File Space Allocation)，又可以用來管理未使用空間？

A screenshot of a white sheet

Description automatically generated

File start 5表示存放檔案從第5個區塊開始，FAT中第5格中的2表示下一個區塊是第2個區塊，以此類推用鏈結串列將檔案存入disk。對於未使用的空間也可以用同樣的方式管理，可依Free找到要用的未使用空間。例如Free 3表示在FAT 3的位置是未使用空間，而FAT 位置3會指向位置6是下一個未使用空間，以此類推

1. 假設一個區塊有512 Bytes，每個索引佔4 Bytes。則1個雙層間接區塊索引(Double Indirect Block Index)可以指向多大資料？

1個index block可放128個索引，每一個索引又指向一個有128個索引的block，一個索引可指向512個資料:

128 \* 128 \* 512 = 8388608 bytes資料

1. 格式化會做哪些事，目的為何ch11 p.28

目的：

物理:磁碟使用前須在每個磁區內寫入記號，用來分辨磁軌及磁區。

邏輯:

產生目錄，以建立磁碟目錄之結構

建立檔案配置表格，以記錄未使用的磁碟空間

找到無法使用的磁碟區塊，避免再使用到

若是開機磁碟，建立置換區域，供記憶體管理系統使用

若是開機磁碟，則要建立開機磁區，供開機使用

1. 連續配置法、連結配置法、及索引配置法，請問何種有內部支離破碎問題？為什麼？何種有外部支離破碎問題？為什麼？

每個檔案空間配置方法都會有內支，因為以區塊為配置單位，檔案最後一個區塊可能發生填不滿block的問題。只有連續配置法會有外支，因為在刪除or生成檔案時，free space會被切的支離破碎。

1. 檔案配置表格(FAT)是屬於那一種檔案空間配置方法？並請說明檔案配置表格的結構ch11 p.17

屬於連結配置法

FAT可以把已使用空間用指標串起來，相對的也可以把未使用的空間用指標記錄起來。

會有一個檔案配置表格紀錄被使用的空間在磁碟中的固定位置且必須是循序執行如圖

例如連結配置法原先在磁碟內的連結情況，可以改用FAT在主記憶體內搜尋連結的情況。